AGWAN: A Generative Model for Labelled, Weighted Graphs

Michael Davis[1], Weiru Liu[1], Paul Miller[1], Ruth F. Hunter[2], and Frank Kee[2]

[1] Centre for Secure Information Technologies,
School of Electronics, Electrical Engineering and Computer Science,
[2] Centre for Public Health, School of Medicine, Dentistry and Biomedical Sciences
Queen's University, Belfast, United Kingdom
{m.davis, w.liu, p.miller, ruth.hunter, f.kee}@qub.ac.uk

Abstract. Real-world graphs or networks tend to exhibit a well-known set of properties, such as heavy-tailed degree distributions, clustering and community formation. Much effort has been directed into creating realistic and tractable models for unlabelled graphs, which has yielded insights into graph structure and evolution. Recently, attention has moved to creating models for labelled graphs: many real-world graphs are labelled with both discrete and numeric attributes. In this paper, we present AGWAN (Attribute Graphs: Weighted and Numeric), a generative model for random graphs with discrete labels and weighted edges. The model is easily generalised to edges labelled with an arbitrary number of numeric attributes. We include algorithms for fitting the parameters of the AGWAN model to real-world graphs and for generating random graphs from the model. Using real-world directed and undirected graphs as input, we compare our approach to state-of-the-art random labelled graph generators and draw conclusions about the contribution of discrete vertex labels and edge weights to graph structure.

Keywords: Network models, graph generators, random graphs, labelled graphs, weighted graphs, graph mining

1 Introduction

Network analysis is concerned with finding patterns and anomalies in real-world graphs, such as social networks, computer and communication networks, or biological and ecological processes. Real graphs exhibit a number of interesting structural and evolutionary properties, such as power-law or log-normal degree distribution, small diameter, shrinking diameter, and the Densification Power Law (DPL) [6, 19, 21].

Besides discovering network properties, researchers are interested in the mechanisms of network formation. Generative graph models provide an abstraction of how graphs form: if the model is accurate, generated graphs will obey the same properties as real graphs. Generated graphs are also useful for simulation experiments, hypothesis testing and making predictions about graph evolution or missing graph elements. Most existing models are for unlabelled, unweighted graphs [6, 19], but some models take discrete vertex labels into account [13, 17, 22]

In this paper, we present AGWAN, a generative model for labelled, weighted graphs. Weights are commonly used to represent the number of occurrences of each edge: the

number of e-mails sent between individuals in a social network [1]; the number of calls to a subroutine in a software call graph [9]; or the number of people walking between a pair of door sensors in a building access control network [8]. In other applications, the edge weight may represent continuous values: donation amounts in a bipartite graph of donors and political candidates [1]; distance or speed in a transportation network [9]; or elapsed time to walk between the sensors in the building network [8]. In some cases, the weight is a multi-dimensional feature vector [8, 9].

Our main motivation for this work is to create a model to better understand the laws governing the relationship between graph structure and numeric labels or weights. Furthermore, we want to be able to create realistic random, labelled, weighted graphs for large-scale simulation experiments for our pattern discovery algorithms [8]. Our experiments in §5 show the extent to which various graph properties are related to labels and weights, and measure exactly how "realistic" our random graphs are. Graphs generated with AGWAN are shown to have more realistic vertex strength distributions and spectral properties than the comparative methods.

This paper is arranged as follows: §2 is an overview of generative graph models; §3 presents AGWAN, our generative model for weighted and numeric labelled graphs. We include a fitting algorithm to learn AGWAN's parameters from a real input graph, and an algorithm to generate random graphs from the model. §4 gives an overview of the datasets that we use in the experiments, and outlines the statistical measures and tests that we use to evaluate the generated graphs. The experiments in §5 demonstrate that the vertex labels and edge weights of a graph can predict the graph structure with high accuracy. Conclusions are in §6.

2 Related Work

Our understanding of the mathematical properties of graph structure was pioneered by Paul Erdős and Alfréd Rényi [10]. Graph formation is modelled as a Bernoulli process, parameterised by the number of vertices and a wiring probability between each vertex pair. While it has been essential to our understanding of component sizes and expected diameter, the Erdős-Rényi model does not explain other important properties of real-world graphs such as degree distribution, transitivity and clustering [6, 21].

Barabási and Albert's Preferential Attachment model [2] uses the "rich get richer" principle to grow graphs from a few vertices up to the desired size. The probability of an edge is proportional to the number of edges already connected to a vertex. This generates graphs with power-law degree distributions. A number of variants of Preferential Attachment have been proposed [6, 21]. Still, Preferential Attachment models lack some desired properties, such as community structure.

The RMat algorithm [7] solves the community structure problem with its recursive matrix approach. RMat graphs consist of 2^n vertices and E edges, with four probabilities a, b, c, d to determine in which quadrant of the adjacency matrix each edge falls. These parameters allow the specification of power-law or log-normal degree distributions; if $a = b = c = d$, the result will be an Erdős-Rényi graph.

Kronecker Graphs [19] fulfil all the properties mentioned above, as well as the DPL and shrinking diameter effect. The model starts with an initiator matrix. Kronecker

multipication is recursively applied to yield the final adjacency matrix of the desired size. This work synthesises the previous work in random graphs in a very elegant way and proves that RMat graphs are a special case of Stochastic Kronecker graphs.

The models above tend to have a small number of parameters and are analytically tractable, with simple and elegant proofs of the desired properties. However, graph labels are not taken into consideration. Stochastic models are another class of generative algorithm which may not be amenable to analytical proofs, but can be fit to real-world labelled graphs and used to learn the properties of those graphs. Models in this category include the Stochastic Block Model [22] and Latent Space approaches [13].

The Multiplicative Attribute Graph (MAG) model [17] draws on both of the above strands of research. MAG is parameterised by the number of vertices, a set of prior probabilities for vertex label values and a set of *affinity matrices* specifying the probability of an edge conditioned on the vertex labels. The affinity matrices can be learned from real graphs using Maximum Likelihood Estimation [16]. [17] proves that Kronecker Graphs are a special case of MAG graphs, and that suitably-parameterized MAG graphs fulfil all the desired properties: log-normal or power-law degree distribution, small diameter, the existence of a unique giant component and the DPL. The MAG model considers discrete vertex labels only. We believe that our method, described in the next section, is the first generative model to include numeric labels or weights.

3 AGWAN: A Generative Model for Labelled, Weighted Graphs

In this section, we present our generative model, AGWAN (Attribute Graph: Weighted and Numeric). The model is illustrated in Fig. 1 for the Enron graph described in §4.

Consider a graph $G = (V, E)$ with discrete vertex label values drawn from a set L. In Fig. 1, $u, v \in V$ are vertices and $w_{uv}, w_{vu} \in \mathbb{R}$ are edge weights. Edges $e \in E$ are specified as a 3-tuple $\langle u, v, w_{uv} \rangle$. In the discussion which follows, we restrict ourselves to a single label on each vertex; we outline how this can be extended to multiple labels in §3.3.

We must choose a suitable probability distribution to model the edge weights accurately and efficiently. The Gaussian distribution is popular as it has an analytically tractable Probability Density Function (PDF). However, the edge weights $W^{ij} = \{w_{ij}\}$ follow an arbitrary probability distribution which is not necessarily Gaussian. By using a weighted mixture of Gaussian components, we can get a reasonable approximation to any general probability distribution [3]. The resulting Gaussian Mixture Model (GMM) is quite flexible and is used extensively in statistical pattern recognition [15].

A parametric GMM can be used where we know the number of components in advance. In our case, the number of components—and therefore the number of parameters in the model—changes according to the data. We avoid the problem of knowing the "correct" number of components by using a non-parametric model. We assume that W^{ij} consists of an infinite number of components and use variational inference to determine the optimal number for our model [4].

The AGWAN model is parameterised by μ, a set of prior probabilities over L; and Θ, a set of edge weight mixture parameters: $\Theta = \{\Omega^{ij} | i, j \in L\}$. For directed graphs, $|\Theta| = |L|^2$ and we need to generate both w_{uv} and w_{vu} (see Fig. 1). For undirected graphs, $\Omega^{ij} = \Omega^{ji}$, so $|\Theta| = O(|L|^2/2)$ and $w_{vu} = w_{uv}$.

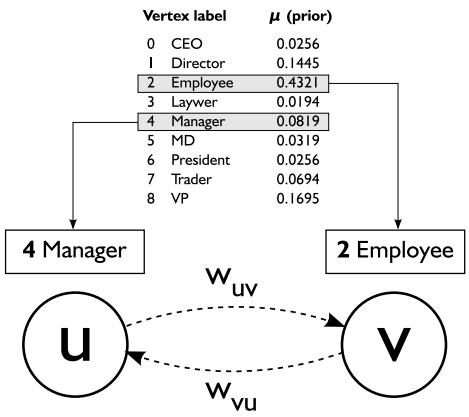


Fig. 1: AGWAN parameters. Vertex labels are selected according to prior probability μ. Edge weight w_{uv} is selected from mixture model Ω^{42} and w_{vu} is selected from mixture model Ω^{24}.

For each combination of vertex attributes $\langle i, j \rangle$, the corresponding mixture model Ω^{ij} parameterises the distribution of edge weights (with an edge weight of 0 indicating no edge). Ω^{ij} is a GMM with M Gaussian components:

$$\Omega^{ij} = \sum_{m=0}^{M-1} \omega_m^{ij} \cdot \eta(\mu_m^{ij}, (\sigma^2)_m^{ij}) \tag{1}$$

where ω_m^{ij} is the weight of each component and $\eta(\mu_m^{ij}, (\sigma^2)_m^{ij})$ is the Gaussian PDF with mean μ_m^{ij} and variance $(\sigma^2)_m^{ij}$. The mixture weights form a probability distribution over the components: $\sum_{m=0}^{M-1} \omega_m^{ij} = 1$. We can specify Ω^{ij} such that the first mixture component encodes the probability of no edge: $\omega_0^{ij} = 1 - P(e_{ij})$, where $P(e_{ij})$ is the probability of an edge between pairs of vertices with labels $\langle i, j \rangle$. The model degenerates to an unweighted graph if there are two components, $\eta_0(0,0)$ and $\eta_1(1,0)$. Furthermore, if the weights ω_m^{ij} are the same for all $\langle i, j \rangle$, the model degenerates to an Erdős-Rényi graph.

As the Gaussian distribution has unbounded support, GMMs can be used to model any set of continuous values. However, if the edge weight is a countable quantity representing the number of occurrences of the edge, then W^{ij} is bounded by $[0, \infty)$. Although this case can be modelled as a GMM, it requires a large number of mixture components to describe the data close to the boundary [20]. We consider alternatives to the GMM for the semi-bounded and bounded cases in §6.

3.1 Graph Generation

Algorithm 1 describes how to generate a random graph using AGWAN(N, L, μ, Θ). The number of vertices in the generated graph is specified by N. After assigning discrete label values to each vertex (lines 2–3, *cf.* Fig. 1), the algorithm checks each vertex pair $\langle u, v \rangle$ for the occurrence of an edge (lines 4–7). If $m = 0$, $\langle u, v \rangle$ is not an edge (line 7). If there is an edge, we assign its weight from mixture component m (lines 8–9). The generated graph is returned as $G = (V, E)$.

Algorithm 1 AGWAN Graph Generation

Require: N (no. of vertices), L (set of discrete label values), μ (prior distribution over L), $\Theta = \{\Omega^{ij}\}$ (set of mixture models)

Create vertex set V of cardinality N, edge set $E = \emptyset$
for all $u \in V$ **do**
    Assign discrete label $l_u \in L$ from prior μ
for all $u, v \in V : u \neq v$ **do**
    $i = l_u, j = l_v$
    Select Gaussian m uniformly at random from Ω^{ij}
    if $m \neq 0$ **then**
        Assign edge weight w_{uv} uniformly at random from $\eta(\mu_m^{ij}, (\sigma^2)_m^{ij})$
        Create edge $e = \langle u, v, w_{uv} \rangle, E = E \cup \{e\}$
return $G = (V, E)$

3.2 Parameter Fitting

To create realistic random graphs, we need to learn the parameters μ, Θ from a real-world input graph G. Let W^{ij} be the set of edge weights between pairs of vertices with labels $\langle i, j \rangle$. During parameter fitting, we want to create a model Ω^{ij} for each W^{ij} in G. Each GMM Ω^{ij} has a finite number of mixture components M. If M is known, Ω^{ij} can be estimated using Expectation Maximisation [12]. However, not only is M unknown, but we expect that it will be different for each Ω^{ij} within a given graph model [8].

We solve this problem by modelling Ω^{ij} as a non-parametric mixture model with an unbounded number of mixture components: a Dirichlet Process Gaussian Mixture Model (DPGMM) [4]. "Non-parametric" does not mean that the model has no parameters; rather, the number of parameters is allowed to grow as more data are observed. In essence, the DPGMM is a probability distribution over the probability distributions of the model.

The Dirichlet Process (DP) over edge weights W^{ij} is a stochastic process $DP(\alpha, H_0)$, where α is a positive scaling parameter and H_0 is a finite measure on W^{ij}; that is, a mapping of the subsets of W^{ij} to the set of non-negative real numbers. If we draw a sample from $DP(\alpha, H_0)$, the result is a random distribution over values drawn from H_0. This distribution H is discrete, represented as an infinite sum of atomic measures. If H_0 is continuous, then the infinite set of probabilities corresponding to the frequency of each possible value that H can return are distributed according to a *stick-breaking process*. The stick-breaking representation of H is given as:

$$\omega_m^{ij}(\mathbf{x}) \prod_{n=1}^{m-1} (1 - \omega_n^{ij}) \qquad\qquad H = \sum_{n=1}^{\infty} \omega_n^{ij}(\mathbf{x}) \delta_{\eta_m^*} \tag{2}$$

where $\{\eta_1^*, \eta_2^*, \ldots\}$ are the atoms representing the mixture components. We learn the mixture parameters using the variational inference algorithm for generating Dirichlet Process Mixtures described in [4]. The weights of each component are generated one-at-a-time by the stick-breaking process, which tends to return the components with the largest weights first. In our experiments, 3–5 mixture components was sufficient to account for over 99% of the data. Mixtures with weights summing to less than 0.01 are dropped from the model, and the remaining weights $\{\omega_m^{ij}\}$ are normalised.

Algorithm 2 AGWAN Parameter Fitting

Require: Input graph $G = (V, E)$

1: $L = \{\text{discrete vertex label values}\}$, $d = |L|$
2: Calculate vertex label priors, apply Laplace smoothing $\forall l \in L : P(l) = \frac{count(l)+\alpha}{N+\alpha d}$
3: $\mu =$ the normalised probability distribution over L such that $\sum_{i=1}^{d} P(l_i) = 1$
4: $\forall i, j \in L : W^{ij} = \emptyset$
5: **for all** $u, v \in V : u \neq v$ **do**
6: $\quad i = l_u, j = l_v$
7: $\quad W^{ij} = W^{ij} \cup \{w_{uv}\}$ ▷ If $\langle u, v \rangle$ is not an edge, then w_{uv} has value zero
8: **for all** $i, j \in L$ **do**
9: $\quad$ estimate Ω^{ij} from W^{ij} using variational inference
10: $\Theta = \{\Omega^{ij}\}$
return μ, Θ

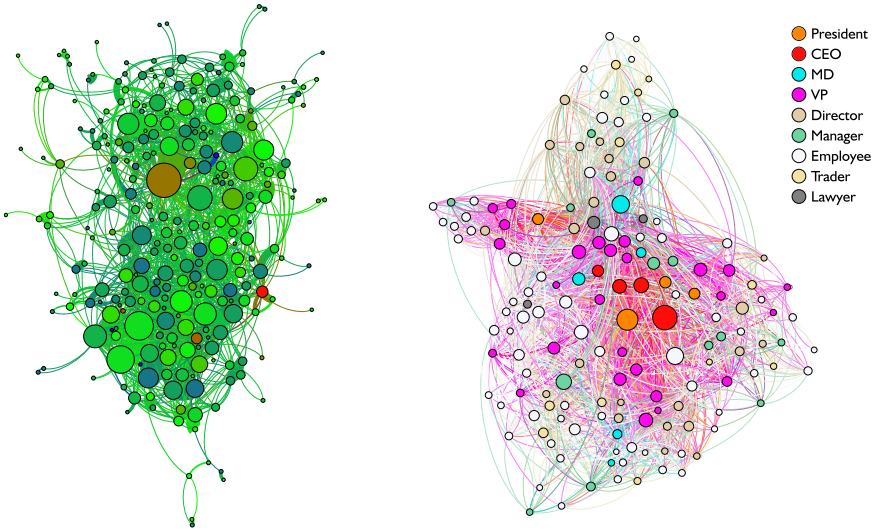
Algorithm 2 is the algorithm for AGWAN parameter fitting. First, we estimate the vertex priors (lines 1–3). Next, we sample the edge weights for each possible combination of vertex label values, with no edge counting as a weight of zero (lines 4–7). Finally, we estimate the GMMs Ω^{ij} from the appropriate set of samples W^{ij} using the the stick-breaking process described above.

3.3 Extending AGWAN to multiple attributes

We have presented AGWAN for a single discrete vertex label and a single numeric edge label (the weight). Many graphs have multiple labels on vertices and edges. AGWAN can be extended to multiple numeric edge labels by generalising the concept of edge weight to k dimensions. In this case, the mean of each mixture component becomes a k-dimensional vector and the variance $(\sigma_m^{ij})^2$ is replaced with the $k \times k$ covariance matrix Σ_m^{ij}. The variational algorithm can be accelerated for higher-dimensional data using a kd-tree [18] and has been demonstrated to work efficiently on datasets of hundreds of dimensions.

A more difficult question is how to extend the model to multiple discrete vertex labels. With even a small number of labels, modelling the full joint probability across all possible combinations of label values becomes a complex combinatorial problem with hundreds or thousands of parameters. The MAG model reduces this complexity by assuming that vertex labels are independent, so edge probabilities can be computed as the product of the probabilities from each label [17]. For latent attributes, MAGFIT enforces independendence by regularising the variational parameters using mutual information [16]. However, the MAG model has not solved this problem for real attributes, where independence cannot be assumed. Furthermore, multiplying the probabilities sets an upper limit (proportional to $\log N$) on the number of attributes which can be used in the model. In our experiments (§5), MAG typically produced the best results with one or two latent variables.

An alternative to multiplying independent probabilities is to calculate the GMM for each edge as the weighted summation of the GMM for each individual attribute. It is likely that some attributes have a large influence on graph structure while others affect it little or not at all. The contribution of each attribute could be estimated using a



(a) **Undirected graph** of who exercised with whom

(b) **Directed graph** of who e-mailed whom

Fig. 2: Input Graph Datasets, from (a) a health study and (b) the Enron e-mail corpus

conditional probability distribution as an approximation to the the joint probability, for example using Markov Random Fields (MRF) or Factor Graphs. This problem remains a topic for further research.

4 Experiments

We evaluate our approach by comparing AGWAN with the state-of-the-art in labelled graph generation, represented by the MAG model [16, 17]. AGWAN and MAG parameters are learned from real-world graphs. We generate random graphs from each model and calculate a series of statistics on each graph. These statistics are used to compare how closely the model maps to the input graph.

Our input datasets are a graph of "who exercised with whom" from a behavioural health study [14] (Fig. 2a, $|V| = 279, |E| = 1308$) and the "who communicates with whom" graph of the Enron e-mail corpus [1] (Fig. 2b, $|V| = 159, |E| = 2667$). Vertices in the health study graph are labelled with 28 attributes representing demographic information and health markers obtained from questionnaire data. Edges are undirected and weighted with the number of mutual coincidences between actors during the study. Vertices in the Enron graph are labelled with the job role of the employee. As e-mail communications are not symmetric, edges are directed and weighted with the number of e-mails exchanged between sender and recipient.

We evaluated AGWAN against the following models:

Erdős-Rényi random graph (ER): The ER model $G(n, p)$ has two parameters. We set the number of vertices n and the edge probability p to match the input graphs as

closely as possible. We do not expect a very close fit, but the ER model provides a useful baseline.

MAG with real attributes (MAG-R1): The MAG model with one real attribute is similar to AGWAN with one real attribute, with the difference that the set of GMMs $\Theta = \{\Omega^{ij}\}$ is replaced with a set of binary edge probabilities, $\Theta = \{p^{ij}\}$.

MAG with latent attributes (MAG-L*x*: The MAG model also allows for modelling the graph structure using latent attributes. The discrete labels provided in the input graph are ignored; instead MAGFIT [16] learns the values of a set of latent attributes to describe the graph structure. To investigate the relative contributions of vertex labels and edge weights to graph structure, we compared MAG models with $x = 1 \ldots 9$ latent binary attributes against AGWAN models with synthetic attributes taking $2^0 \ldots 2^9$ values.

As ER and MAG do not generate weighted graphs, we set the weight of the edges in the generated graphs to the mean edge weight from the input graphs. This ensures that statistics such as average vertex strength are not skewed by unweighted edges.

To evaluate the closeness of fit of each model, we use the following statistics:

Vertex Strength: For an unweighted graph, one of the most important measures is the degree distribution (the number of in-edges and out-edges of each vertex). Real-world graphs tend to have heavy-tailed power-law or log-normal degree distributions [6, 21]. For a weighted graph, we generalise the concept of vertex degree to vertex strength [11]:

$$s_u = \sum_{v \neq u} w_{uv} \tag{3}$$

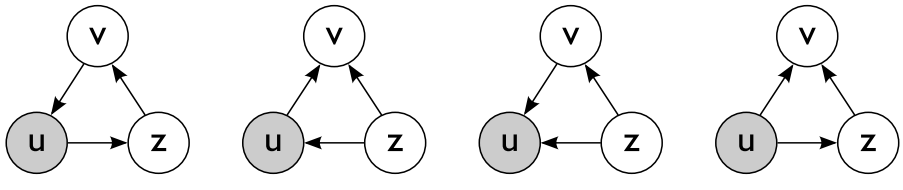
For the undirected graphs, we plot the Complementary Cumulative Distribution Function (CCDF) of the total strength of each vertex. For the directed graphs, we plot the CCDFs for in-strength and out-strength.

Spectral Properties: We use Singular Value Decomposition (SVD) to calculate the singular values and singular vectors of the graph's adjacency matrix, which act as a signature of the graph structure. In an unweighted graph, the adjacency matrix contains binary values, for "edge" or "no edge". In a weighted graph, the adjacency matrix contains the edge weights (with 0 indicating no edge). For SVD $U\Sigma V$, we plot Cumulative Distribution Functions (CDFs) of the singular values Σ and the components of the left singular vector U corresponding to the highest singular value.

Clustering Coefficients: the clustering coefficient C is an important measure of community structure. It measures the density of triangles in the graph, or the probability that two neighbours of a vertex are themselves neighbours [21]. We extend the notion of clustering coefficients to weighted, directed graphs using the equation in [11]:

$$C_u = \frac{[\mathbf{W}_u^{[\frac{1}{3}]} + (\mathbf{W}_u^T)^{[\frac{1}{3}]}]^3_{uu}}{2[d_u^{tot}(d_u^{tot} - 1) - 2d_u^{\leftrightarrow}]} \tag{4}$$

where C_u is the weighted clustering coefficient for vertex u, $\mathbf{W}_u$ is the weighted adjacency matrix for u and its neighbours, $\mathbf{W}^T$ is the transpose of $\mathbf{W}$, d_u^{tot} is the total degree of a vertex (the sum of its in- and out-degrees) and $d_u^{\leftrightarrow}$ is the number of bilateral edges in u (the number of neighbours of u which have both an in-edge and an out-edge between themselves and u).



(a) Cycle (b) Middleman (c) In (d) Out

Fig. 3: Triad Patterns in a Directed Graph

Triad Participation: Closely related to the clustering coefficient is the concept of triangle or triad participation. The number of triangles that a vertex is connected to is a measure of transitivity [21]. For the directed graphs, the triangles have a different interpretation depending on the edge directions. There are four types of triangle pattern [11], as shown in Fig. 3. To generalise the concept of triad participation to weighted, directed graphs, we consider each of the four triangle types separately, and sum the total strength of the edges in each triad:

$$t^y_u = \sum_{v,z \in \mathbf{W}_u \setminus u} \mathbb{W}^y_{uvz} \tag{5}$$

where $y = \{cycle, middleman, in, out\}$ is the triangle type and $\mathbb{W}^y_{uvz}$ is calculated as shown in Fig. 3 for each triangle type y.

To give a more objective measure of the closeness of fit between the generated graphs and the input graph, we use a Kolmogorov-Smirnov (KS) test and the L2 (Euclidean) distance between the CDFs for each statistic. As the CDFs are for heavy-tailed distributions, we use the logarithmic variants of these measures [16]. The KS and L2 statistics are calculated as:

$$KS(D_1, D_2) = max_x |\log D_1(x) - \log D_2(x)| \tag{6}$$

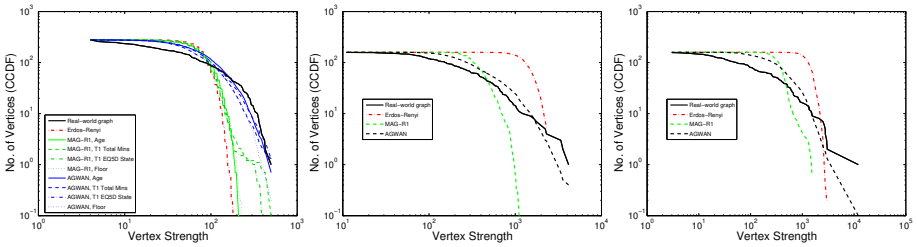
$$L2(D_1, D_2) = \sqrt{\frac{1}{\log b - \log a} \sum_{x=a}^{b} (\log D_1(x) - \log D_2(x))^2} \tag{7}$$

where $[a, b]$ is the interval for the support of distributions D_1 and D_2.

The model that generates graphs with the lowest KS and L2 values for each of the statistics discussed above has the closest fit to the real-world graph.

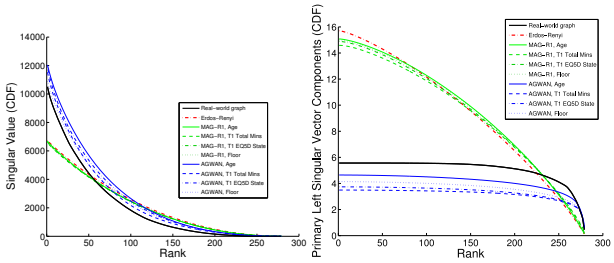
5 Results

For each model, we generated 10 random graphs and calculated statistics for each. The plots of the averaged CDFs of the 10 graphs for each model are shown in Figs. 4–11. Tables for the closeness of fit of each CDF (KS and L2 statistics) are in the appendix.

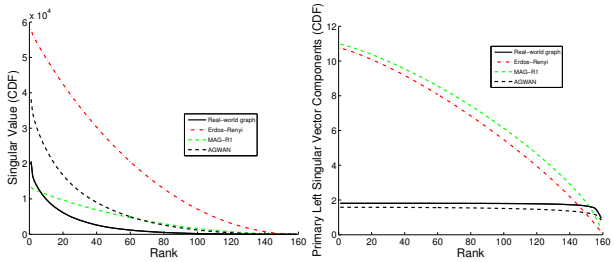


(a) Undirected (b) Directed (In-strength) (c) Directed (Out-strength)

Fig. 4: Vertex Strength Distribution—Real Attributes

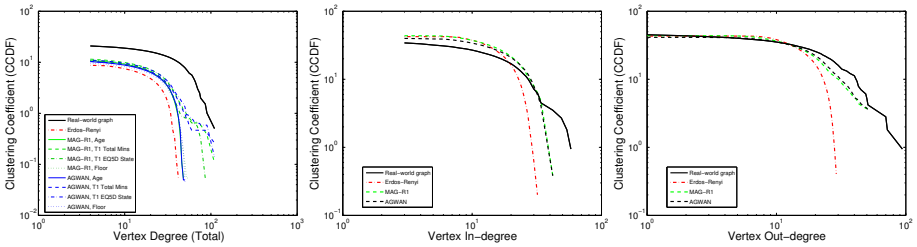


(a) Undirected—Singular Values (b) Undirected—Primary Left Singular Vector



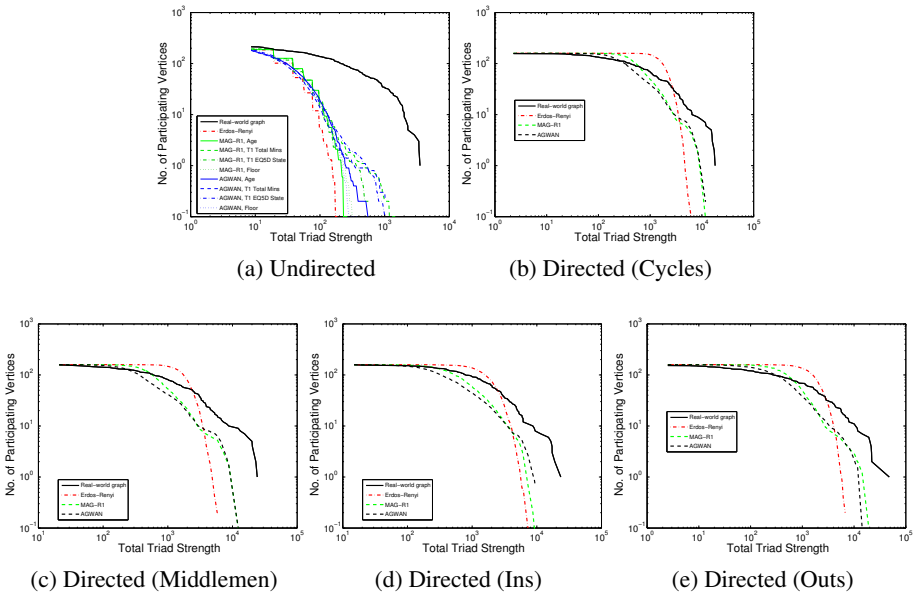
(c) Directed—Singular Values (d) Directed—Primary Left Singular Vector

Fig. 5: Spectral Properties—Real Attributes



(a) Undirected (b) Directed (In-edges) (c) Directed (Out-edges)

Fig. 6: Clustering Coefficients—Real Attributes



(a) Undirected (b) Directed (Cycles)

(c) Directed (Middlemen) (d) Directed (Ins) (e) Directed (Outs)

Fig. 7: Triad Participation—Real Attributes

5.1 Real Attributes

For the undirected graph (Health Study, Fig. 2a), we show results for four vertex attributes: age; total minutes spent exercising; EQ5D State (a quality-of-life metric determined by questionnaire); and Floor (the building and floor number where the person works; people who work on the same floor were highly likely to exercise together). For the directed graph (Enron, Fig. 2b), we have one vertex attribute, the person's job role.

Vertex Strength (Fig. 4): The graphs generated from AGWAN have vertex strength distributions which map very closely to the input graphs. The graphs generated from MAG-R1 are better than random (ER), but the vertex strength distribution is compressed into the middle part of the range, with too few high- and low-strength vertices. This indicates that vertex strength depends on both the label distribution and the edge weight distribution; AGWAN models both of these, whereas MAG models only the former.

Spectral Properties (Fig. 5): The spectral properties of the AGWAN graphs map very closely to the input graphs. The singular values follow the same curve as the input graphs, indicating that graphs generated with AGWAN have similar connectivity to the input graph [6]. The primary singular vector components also follow the same shape and map very closely to the input graph. For MAG-R1, the singular values follow a straight line rather than a curve, because MAG does not model the edge weight distribution. The primary singular vector components are no better than random, because it is not possible to accurately model singular vectors without taking the edge weights into account.

Clustering Coefficients (Fig. 6): The accuracy of AGWAN and MAG-R1 is similar; better than random but not as close a fit as for the first two statistics. The results for vertex strength and spectral properties did not strongly depend on which attribute was

chosen, but here it makes a difference: Total Mins and EQ5D State give better results than Age and Floor. This implies that some attributes can predict community formation better than others. As the results for both approaches are similar, we conclude that the processes that give rise to clustering are independent of the edge weight distribution.

Triad Participation (Fig. 7): As triad participation is closely related to clustering, it is no surprise that the results are comparable: the accuracy of AGWAN and MAG-R1 is similar; better than random, but not as close as for vertex strength and spectral properties. Triad participation appears to be dependent to some extent on vertex label values but independent of the edge weight distribution.

One of the findings in [16] was that clustering arises from multiple processes (homophily and core-periphery). "Simplified MAG" (where all attributes are the same) could not model the clustering property, implying that it is not possible to accurately reproduce clustering when the model has only one attribute. We propose to extend our model to more than one attribute as outlined in §3.3 to investigate whether this produces a more accurate model of clustering and triad participation.

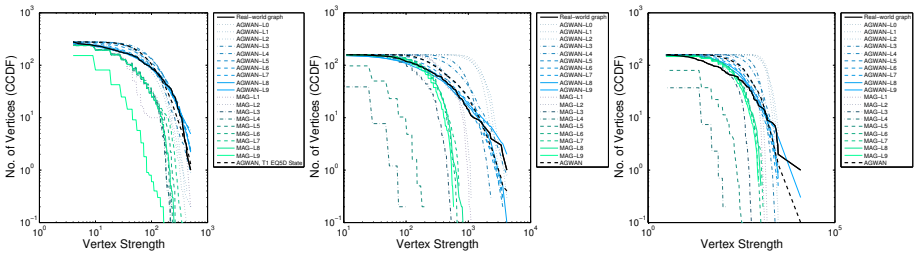
5.2 Synthetic Attributes

An alternate interpretation of the MAG model ignores the true attribute values from the input graph and represents attributes as latent variables, which are learned using a variational inference EM approach [16]. To compare AGWAN with this approach, we replaced the real labels in the input graph with a synthetic vertex attribute taking $2^0 \dots 2^9$ values allocated uniformly at random, then learned the edge weight distributions using variational inference as normal. We have plotted AGWAN with one real attribute alongside for comparison.

Vertex Strength (Fig. 8): AGWAN with synthetic attributes has similar accuracy to AGWAN-R1. Varying the number of synthetic attributes has a small effect on the accuracy. MAG with latent attributes has similar accuracy to MAG-R1. Varying the number of synthetic attributes causes a large variation in the accuracy. We conclude that vertex strength is dependent on both edge weight and vertex label distribution, but the edge weights play a more important role.

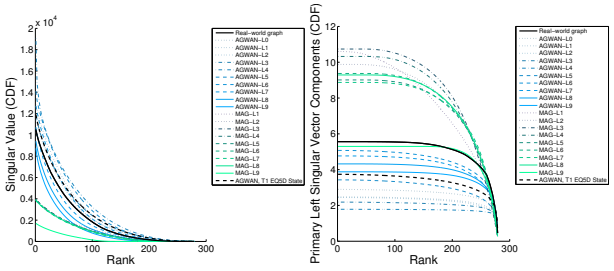
Spectral Properties (Fig. 9): For AGWAN, the spectral properties follow the same curves as the input graphs. For singular values, varying the number of synthetic attributes causes a small variation in the closeness of fit. For singular vectors, the accuracy is highly dependent on the number of synthetic attributes. For MAG, the singular values are almost a straight line, as the edge weight distribution is not taken into account. The singular vectors in general do not match very closely. It is possible to get a good fit using many latent attributes, but this compromises the other statistics which fit better with few latent attributes. We conclude that spectral properties are dependent on both edge weight and vertex label distribution.

Clustering Coefficients (Fig. 10): Both approaches are significantly more accurate using synthetic attributes than they were with real attributes. This implies that while real labels are influenced by the (unobserved) process which gives rise to clustering, synthetic labels with more degrees of freedom can model it more accurately. As before, clustering appears to be independent of the edge weight distribution.



(a) Undirected (b) Directed (In-strength) (c) Directed (Out-strength)

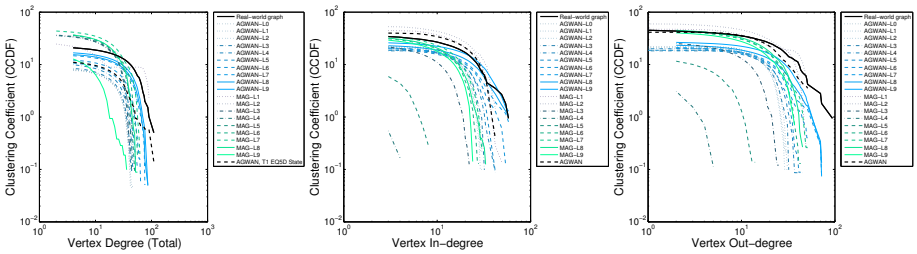
Fig. 8: Vertex Strength Distribution—Synthetic Attributes



(a) Undirected—Singular Values (b) Undirected—Primary Left Singular Vector

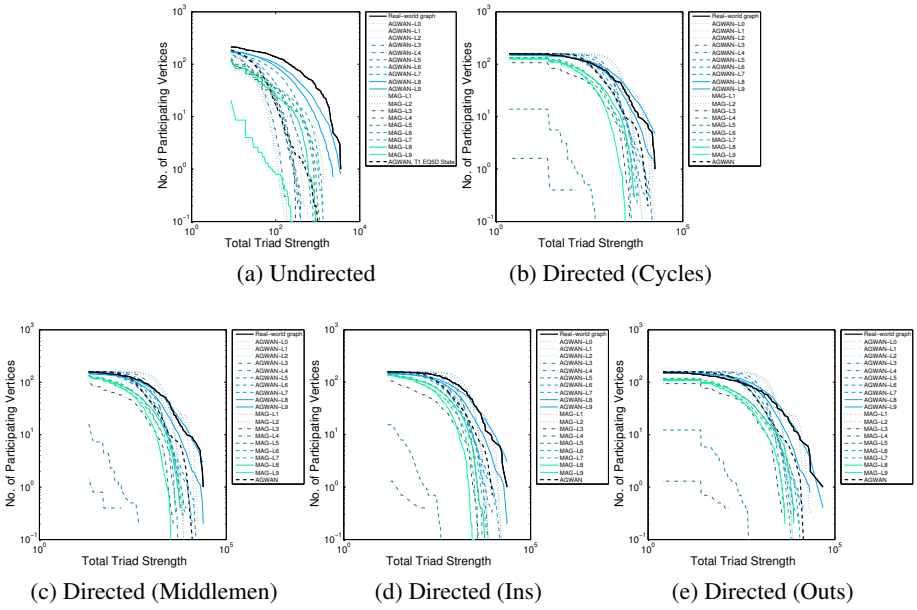
(c) Directed—Singular Values (d) Directed—Primary Left Singular Vector

Fig. 9: Spectral Properties—Synthetic Attributes



(a) Undirected (b) Directed (In-edges) (c) Directed (Out-edges)

Fig. 10: Clustering Coefficients—Synthetic Attributes



(a) Undirected (b) Directed (Cycles)

(c) Directed (Middlemen) (d) Directed (Ins) (e) Directed (Outs)

Fig. 11: Triad Participation—Synthetic Attributes

Triad Participation (Fig. 11): As with clustering, synthetic vertex labels can model the process that gives rise to triad participation, while edge weights have little or no influence.

In general, MAG acheives the best results when there are one or two vertex attributes, whereas AGWAN performs best when there are 7 or 8 attributes. MAG assumes that each attribute is independent, so there is a limit on the number of attributes that can be included in the model (proportional to $\log N$). Above this limit, the performance of the model degrades. With AGWAN, there is no independence assumption, so the attributes model the full joint probability. As the number of attribute values (2^x) approaches N, there is a danger of overfitting and the model performance degrades.

6 Conclusions

We presented AGWAN, a model for random graphs with discrete labels and weighted edges. We included a fitting algorithm to learn a model of graph edge weights from real-world data, and a generative algorithm to generate random labelled, weighted graphs with similar characteristics to the real-world graph.

We measured the closeness of fit of our generated graphs to the input graph over a range of graph statistics, and compared our approach to the state-of-the-art in random graph generative algorithms. Our results demonstrate that AGWAN produces an accurate model of the properties of a weighted real-world graph. For vertex strength distribution and spectral properties, AGWAN is shown to produce a closer fit than MAG.

For clustering and triad participation, we achieved a closer fit using synthetic attributes than using real attributes. This is consistent with the results for MAG for unweighted graphs [17]. Further research is required into the relationship between vertex attributes and triangle formation in graphs; our results indicate that edge weights do not play an important part in these processes. We propose to extend AGWAN to multiple vertex labels to investigate the effect on clustering.

In §3, we considered the case where edge weights are countable quantities bounded by $[0, \infty)$. As GMMs are unbounded, it may be more appropriate to model the edge weights using a truncated GMM [20] or Beta Mixture Model [5]. We propose to investigate these alternatives in future work.

As discussed in §3.3, MAG's method of combining multiple vertex attributes is unsatisfactory when applied to real attributes, due to the assumption of independence and the limit on the number of attributes which can be modelled. We have proposed a future line of research based on a weighted summation of the GMM for each edge. The fitting algorithm would need to regularise the individual contributions of each edge to take account of dependencies. The complexity of modelling the full joint distribution could be reduced with an approach based on Markov Random Fields or Factor Graphs.

References

1. Akoglu, L., McGlohon, M., Faloutsos, C.: OddBall: Spotting anomalies in weighted graphs. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD (2). Lecture Notes in Computer Science, vol. 6119, pp. 410–421. Springer (2010)
2. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science 286(5439), 509–512 (1999)
3. Bishop, C.M.: Pattern Recognition and Machine Learning. Information Science and Statistics, Springer, New York, USA, 3rd edn. (2011)
4. Blei, D.M., Jordan, M.I.: Variational inference for Dirichlet process mixtures. Bayesian Analysis 1, 121–144 (2005)
5. Bouguila, N., Ziou, D., Monga, E.: Practical Bayesian estimation of a finite Beta mixture through Gibbs sampling and its applications. Statistics and Computing 16(2), 215–225 (2006)
6. Chakrabarti, D., Faloutsos, C.: Graph Mining: Laws, Tools, and Case Studies. Synthesis Lectures on Data Mining and Knowledge Discovery, Morgan & Claypool Publishers (2012)
7. Chakrabarti, D., Zhan, Y., Faloutsos, C.: R-MAT: A recursive model for graph mining. In: Berry, M.W., Dayal, U., Kamath, C., Skillicorn, D.B. (eds.) SDM. SIAM (2004)
8. Davis, M., Liu, W., Miller, P.: Finding the most descriptive substructures in graphs with discrete and numeric labels. Journal of Intelligent Information Systems pp. 1–26 (Dec 2013)
9. Eichinger, F., Huber, M., Böhm, K.: On the usefulness of weight-based constraints in frequent subgraph mining. In: Bramer, M., Petridis, M., Hopgood, A. (eds.) SGAI Conf. pp. 65–78. Springer (2010)
10. Erdős, P., Rényi, A.: On the evolution of random graphs. Publication of the Mathematical Institute of the Hungarian Academy of Sciences 5, 17–61 (1960)
11. Fagiolo, G.: Clustering in complex directed networks. Physical Review E 76(2) (Aug 2007)
12. Figueiredo, M.A.T., Jain, A.K.: Unsupervised learning of finite mixture models. IEEE Trans. Pattern Anal. Mach. Intell. 24(3), 381–396 (2002)
13. Hoff, P.D., Raftery, A.E., Handcock, M.S.: Latent space approaches to social network analysis. Journal of the American Statistical Association 97(460), 1090–1098 (Dec 2002)

14. Hunter, R.F., Davis, M., Tully, M.A., Kee, F.: The physical activity loyalty card scheme: Development and application of a novel system for incentivizing behaviour change. In: Kostkova, P., Szomszor, M., Fowler, D. (eds.) eHealth. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 91, pp. 170–177. Springer (2011)
15. Jain, A.K., Duin, R.P.W., Mao, J.: Statistical pattern recognition: A review. IEEE Trans. Pattern Anal. Mach. Intell. 22(1), 4–37 (2000)
16. Kim, M., Leskovec, J.: Modeling social networks with node attributes using the Multiplicative Attribute Graph model. In: Cozman, F.G., Pfeffer, A. (eds.) UAI. pp. 400–409. AUAI Press (2011)
17. Kim, M., Leskovec, J.: Multiplicative Attribute Graph model of real-world networks. Internet Mathematics 8(1–2), 113–160 (2012)
18. Kurihara, K., Welling, M., Vlassis, N.A.: Accelerated variational Dirichlet process mixtures. In: Schölkopf, B., Platt, J.C., Hoffman, T. (eds.) NIPS. pp. 761–768. MIT Press (2006)
19. Leskovec, J., Chakrabarti, D., Kleinberg, J.M., Faloutsos, C., Ghahramani, Z.: Kronecker graphs: An approach to modeling networks. Journal of Machine Learning Research 11, 985–1042 (2010)
20. Lindblom, J., Samuelsson, J.: Bounded support Gaussian mixture modeling of speech spectra. IEEE Transactions on Speech and Audio Processing 11(1), 88–99 (2003)
21. Newman, M.: Networks: An Introduction. OUP, New York, NY, USA (2010)
22. Wang, Y., Wong, G.: Stochastic block models for directed graphs. Journal of the American Statistical Association 82(397), 8–19 (1987)

Appendix: KS and L2 Statistics

	E-R	MAG-R1				AGWAN			
		Age	Total Mins	EQ5D State	Floor	Age	Total Mins	EQ5D State	Floor
Vertex Strength	6.064	5.940	2.957	3.689	5.799	0.799	1.081	**0.635**	1.674
Singular Values	36.193	35.644	35.393	35.612	36.001	34.482	**32.319**	33.720	34.946
Singular Vector	1.323	1.239	0.964	0.984	1.134	**0.248**	0.491	0.450	0.371
Clustering Coefficient	5.224	5.048	2.083	3.343	4.895	5.132	2.493	**2.042**	5.161
Triad Participation	7.012	6.877	5.704	5.704	6.685	6.328	**5.106**	5.829	6.768

Table 1: KS Statistic for Undirected Graph, Real Attributes (Figs. 4–7)

	E-R	MAG-R1				AGWAN			
		Age	Total Mins	EQ5D State	Floor	Age	Total Mins	EQ5D State	Floor
Vertex Strength	9.686	7.281	8.265	9.377	10.039	1.829	2.589	**1.765**	3.294
Singular Values	41.815	41.298	41.052	41.227	41.623	39.629	**38.211**	39.100	40.060
Singular Vector	5.004	4.940	4.614	4.742	4.852	**1.486**	3.257	2.914	2.307
Triad Participation	**16.879**	17.334	18.828	18.861	17.101	19.746	19.434	18.348	20.288

Table 2: L2 Statistic for Undirected Graph, Real Attributes (Figs. 4–7)

	E-R	MAG-R1	AGWAN
In-Vertex Strength	2.469	4.700	**1.455**
Out-Vertex Strength	2.708	2.659	**2.303**
Singular Values	37.235	35.752	**34.894**
Singular Vector	1.915	1.801	**0.282**
Clustering Coefficient (In-Edges)	3.444	**2.208**	2.220
Clustering Coefficient (Out-Edges)	3.728	0.769	**0.702**
Clustering Coefficient	4.347	**1.651**	3.163
Triad Participation (Cycles)	4.787	4.248	**3.555**
Triad Participation (Middlemen)	4.382	**4.500**	**4.500**
Triad Participation (Ins)	4.700	4.500	**2.436**
Triad Participation (Outs)	4.382	**4.094**	4.248

Table 3: KS Statistic for Directed Graph, Real Attributes (Figs. 4–7)

	E-R	MAG-R1	AGWAN
In-Vertex Strength	5.679	4.912	**1.816**
Out-Vertex Strength	5.100	3.534	**2.117**
Singular Values	25.044	19.546	**18.360**
Singular Vector	7.316	7.587	**0.988**
Clustering Coefficient (In-Edges)	3.528	1.607	**1.528**
Clustering Coefficient (Out-Edges)	3.145	1.191	**1.002**
Clustering Coefficient	6.949	**1.438**	2.284
Triad Participation (Cycles)	3.823	**3.000**	3.101
Triad Participation (Middlemen)	5.144	**4.178**	4.207
Triad Participation (Ins)	4.630	4.826	**4.332**
Triad Participation (Outs)	3.727	3.295	**3.203**

Table 4: L2 Statistic for Directed Graph, Real Attributes (Figs. 4–7)

	MAG Latent									AGWAN
	1	2	3	4	5	6	7	8	9	EQ5D State
Vertex Strength	**2.243**	5.106	5.886	5.886	5.670	5.481	4.605	5.561	6.234	0.635
Singular Values	**30.901**	46.771	89.148	93.658	81.082	93.413	125.855	72.059	85.863	33.720
Singular Vector	0.645	0.654	0.821	0.694	0.590	0.561	0.645	0.579	**0.313**	0.450
Clustering Coefficient	**1.283**	4.406	3.863	4.575	4.401	3.470	3.256	4.397	4.773	2.042
Triad Participation	**3.829**	6.292	6.709	6.593	6.016	5.768	4.868	5.914	6.877	5.829

	AGWAN									
	0	1	2	3	4	5	6	7	8	9
Vertex Strength	3.401	2.197	2.303	1.050	1.758	0.916	0.975	0.875	**0.854**	1.589
Singular Values	35.238	35.194	35.226	35.341	35.542	33.763	32.824	**27.713**	34.052	37.384
Singular Vector	0.675	0.827	0.847	0.950	1.139	0.559	**0.183**	0.221	0.258	0.361
Clustering Coefficient	5.353	5.350	3.561	4.615	4.395	4.054	4.470	3.676	**3.401**	3.440
Triad Participation	6.985	7.090	6.994	5.991	5.872	6.607	6.131	5.561	2.238	**1.204**

Table 5: KS Statistic for Undirected Graph, Synthetic Attributes (Figs. 8–11)

	MAG Latent									AGWAN
	1	2	3	4	5	6	7	8	9	EQ5D State
Vertex Strength	**7.944**	8.473	9.236	10.783	10.103	8.635	9.120	9.603	21.027	1.765
Singular Values	**55.080**	94.881	106.265	109.813	104.160	109.673	120.108	113.166	173.884	39.100
Singular Vector	3.231	3.324	3.895	3.622	2.894	3.092	2.873	3.079	**0.396**	2.914
Triad Participation	12.047	15.550	15.821	17.494	11.038	11.646	**10.367**	14.507	29.136	18.348

	AGWAN									
	0	1	2	3	4	5	6	7	8	9
Vertex Strength	6.266	4.537	3.754	2.584	2.160	1.731	1.343	0.873	**0.693**	1.229
Singular Values	40.448	40.394	40.391	40.504	40.873	38.980	37.613	**27.296**	44.148	74.019
Singular Vector	4.477	5.513	5.671	6.316	7.530	3.612	**0.866**	1.237	1.719	2.351
Triad Participation	22.841	20.975	23.682	17.878	17.287	16.174	15.254	10.310	5.753	**2.803**

Table 6: L2 Statistic for Undirected Graph, Synthetic Attributes (Figs. 8–11)

	MAG Latent									AGWAN
	1	2	3	4	5	6	7	8	9	Employee Type
In-Vertex Strength	4.700	**3.602**	5.991	6.522	6.142	5.704	3.951	5.347	5.193	1.455
Out-Vertex Strength	4.942	4.605	5.768	5.991	6.234	5.075	4.317	3.466	**3.401**	2.303
Singular Values	35.715	35.591	27.492	89.063	148.080	32.392	**1.708**	31.555	37.163	34.894
Singular Vector	1.636	1.630	1.453	**0.190**	0.765	1.586	1.525	1.526	1.552	0.282
Clustering Coefficient (In-Edges)	2.961	**0.897**	4.775	5.294	4.578	4.357	3.302	3.770	4.512	2.220
Clustering Coefficient (Out-Edges)	3.164	**0.513**	5.193	5.877	5.463	4.363	3.142	3.273	2.865	0.702
Clustering Coefficient	3.278	**2.347**	5.251	6.255	5.839	4.387	3.739	4.339	4.000	3.163
Triad Participation (Cycles)	3.912	**2.996**	5.347	5.940	6.867	5.247	4.094	3.843	5.704	3.555
Triad Participation (Middlemen)	4.248	**3.401**	4.942	5.920	6.319	4.339	3.602	3.689	5.858	4.500
Triad Participation (Ins)	**3.912**	**3.912**	5.670	5.940	7.170	5.704	4.700	5.075	6.153	2.436
Triad Participation (Outs)	**1.476**	2.526	4.571	5.695	6.768	5.075	4.500	4.094	4.745	4.248

	AGWAN									
	0	1	2	3	4	5	6	7	8	9
In-Vertex Strength	2.418	2.513	2.345	2.590	**1.120**	2.303	1.897	2.015	2.303	0.693
Out-Vertex Strength	2.996	2.234	2.090	4.248	**1.122**	1.150	1.514	1.966	1.386	1.204
Singular Values	37.497	37.866	37.377	36.590	36.159	34.801	33.812	32.696	**26.494**	8.327
Singular Vector	1.887	1.962	1.811	1.665	**0.616**	1.130	0.824	0.908	0.887	0.789
Clustering Coefficient (In-Edges)	3.477	3.567	4.386	4.159	3.704	3.682	2.678	0.662	**0.460**	0.492
Clustering Coefficient (Out-Edges)	4.945	4.316	5.134	4.969	4.948	4.747	**2.563**	3.200	2.605	3.204
Clustering Coefficient	4.580	4.018	4.837	2.691	4.369	3.933	1.501	**1.075**	2.620	0.848
Triad Participation (Cycles)	4.500	4.500	2.659	3.912	3.602	3.283	2.996	3.912	**1.204**	1.548
Triad Participation (Middlemen)	4.787	4.787	4.094	5.247	3.843	3.314	3.807	4.248	**1.609**	1.099
Triad Participation (Ins)	4.700	4.700	4.007	5.298	4.700	3.283	2.862	4.094	**1.609**	1.099
Triad Participation (Outs)	4.942	4.942	3.624	4.094	3.977	3.912	3.114	2.862	**1.696**	0.916

Table 7: KS Statistic for Directed Graph, Synthetic Attributes (Figs. 8–11)

	MAG Latent									AGWAN
	1	2	3	4	5	6	7	8	9	Employee Type
In-Vertex Strength	5.023	**3.055**	8.856	19.820	15.718	8.678	6.171	8.672	7.066	1.816
Out-Vertex Strength	3.001	3.704	7.805	14.329	10.882	3.740	3.120	**2.668**	3.737	2.117
Singular Values	19.285	18.938	13.768	90.672	160.831	28.601	**6.158**	28.074	38.490	18.360
Singular Vector	7.470	7.530	7.100	**0.388**	4.062	7.453	7.200	7.266	7.339	0.988
Clustering Coefficient (In-Edges)	2.507	**1.786**	6.733	12.533	7.692	5.841	4.184	5.705	4.819	1.528
Clustering Coefficient	2.450	**2.419**	10.611	22.886	13.922	5.851	4.568	5.381	7.653	2.284
Triad Participation (Cycles)	2.060	**1.800**	7.788	15.981	16.270	6.781	6.121	5.378	8.763	3.101
Triad Participation (Middlemen)	2.828	**1.771**	11.094	19.126	18.575	7.016	7.204	6.517	11.150	4.207
Triad Participation (Ins)	3.293	**1.902**	11.473	12.061	16.361	9.756	8.905	9.124	13.740	4.332
Triad Participation (Outs)	**1.459**	1.816	6.646	17.093	14.603	5.950	5.399	4.698	6.315	3.203

	AGWAN									
	0	1	2	3	4	5	6	7	8	9
In-Vertex Strength	5.638	5.774	5.473	4.355	3.151	2.071	1.367	**1.299**	1.412	0.665
Out-Vertex Strength	5.128	4.807	4.732	4.756	3.060	2.224	1.918	2.034	**1.415**	1.045
Singular Values	25.020	25.815	24.922	22.017	20.767	18.270	16.748	15.010	**12.516**	8.758
Singular Vector	7.814	6.764	7.798	5.949	**2.643**	5.471	4.088	4.421	2.725	1.396
Clustering Coefficient (In-Edges)	3.987	4.972	5.834	4.314	3.846	3.413	2.575	1.524	**0.999**	0.686
Clustering Coefficient	7.065	8.188	9.244	6.606	7.581	6.872	4.951	**4.189**	3.658	2.536
Triad Participation (Cycles)	3.212	3.017	2.407	4.816	3.728	3.856	3.566	3.733	**1.113**	1.014
Triad Participation (Middlemen)	4.670	4.310	3.586	7.121	5.734	5.924	5.288	4.942	**2.382**	0.611
Triad Participation (Ins)	4.391	3.757	3.575	7.742	6.376	6.616	5.902	5.306	**2.464**	0.936
Triad Participation (Outs)	4.887	4.537	3.305	4.615	4.540	4.963	4.359	3.978	**1.947**	0.589

Table 8: L2 Statistic for Directed Graph, Synthetic Attributes (Figs. 8–11)